Downloaded from orbit.dtu.dk on: May 06, 2023

DTU Library

An exact approach for aggregated formulations

Gamst, Mette; Spoorendonk, Simon

Publication date:
2013

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):
Gamst, M., & Spoorendonk, S. (2013). *An exact approach for aggregated formulations*. DTU Management Engineering. DTU Management Engineering Report No. 3.2013

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

An exact approach for aggregated formulations

Report 3.2013

DTU Management Engineering

Mette Gamst
Simon Spoorendonk

March 2013

DTU Management Engineering
Department of Management Engineering

An exact approach for aggregated formulations

Mette Gamst* Simon Spoorendonk*

March 7, 2013

Abstract

Aggregating formulations is a powerful approach for transforming problems into taking more tractable forms. Aggregated formulations can, though, have drawbacks: some information may get lost in the aggregation and – put in a branch-and-bound context – branching may become very difficult and even cause an infeasible solution.

In this paper, we consider (mixed) integer program formulations and propose a method for ensuring an optimal solution to the original (disaggregated) problem using an aggregated formulation. The method is based on Benders' decomposition on a combination of the disaggregated mathematical formulation and the aggregated formulation. The method allows usage of relaxed aggregated formulations and enables branching on both aggregated and disaggregated variables. Also, the method guarantees an LP bound at least as good as those for the disaggregated and aggregated formulations.

The paper includes general considerations on types of problems for which the method is of particular interest. Furthermore, we prove the correctness of the procedure and consider how to include extensions such as cutting planes and advanced branching strategies.

1 Introduction

Aggregating (mixed) integer program formulations is a popular modeling approach for reaching simpler and more tractable mathematical formulations. Variable aggregation is among others performed as part of Dantzig-Wolfe decomposition [8]. Algorithms such as branch-and-price and branch-and-cut-and-price have gained much attention in state-of-the-art research as they tend to show superior performance across many applications [15]. Aggregation may lead to mathematical formulations with a different solution polytope than that for the original (disaggregated) formulation, and the aggregated formulation may be a *relaxation* of the original problem. In a branch-and-bound context, aggregation can also lead to a formulation where

*DTU Management Engineering, Technical University of Denmark, Denmark; e-mail: {gamst,spoo}@man.dtu.dk

branching is not trivial, for example when optimality cannot be guaranteed by branching on the aggregated variables. Examples of problems, where aggregation has such side effects are:

- The Split Delivery Vehicle Routing Problem (SDVRP) introduced by Dror and Trudeau [9] is a vehicle routing problem, where customers have a certain demand, which can be satisfied through multiple visits. The disaggregated formulation contains variables for each edge and vehicle and for each customer and vehicle [12, 14]. It is difficult to solve to optimality due to symmetry in the solution polytope. An alternative approach in the literature is to solve an aggregated formulation containing only variables for edges and for customers [5]. The latter is faster to solve, but it is a relaxation: feasible solutions to the aggregated formulation are not always feasible to the original formulation.

- The Cutting Stock problem (CS) was introduced by Kantorovich [13] and consists in cutting a given number of items on stocks subject to item and stock lengths such that the number of used stocks is minimized. The CS is typically modeled using the Gilmore-Gomory formulation [11], which consists of pattern variables and takes on a set cover-like form. Column generation is used for generating pattern variables. Branching on the variables in the Gilmore-Gomory formulation complicates the structure of the pricing problem from the tractable integer knapsack problem to the more difficult constrained shortest path problem. This slows down the column generation process [1]. The disaggregated formulation for the CS is an arc-flow formulation, which contains much symmetry and is thus less tractable.

- The k-splittable flow problem (kFP) was presented by Baier et al. [4] and consists in routing a commodity through a capacitated network using at most k paths. The disaggregated formulation keeps track of flow on each of the k paths, i. e. it contains a path index on variables [18]. This causes symmetry in the solution polytope and a large number of variables and constraints. An alternative approach is to perform Dantzig-Wolfe decomposition such that the path index is left out [10]. While the latter approach generally performs better, branching on the aggregated variables is difficult and produces large search trees.

Generic branching in aggregated formulations is a well-known challenge in operations research, and almost all work on aggregated formulations uses application specific branching, see e. g. [6, 18, 19]. Some more generic branching strategies have been proposed, though. Villeneuve et al. [21] consider branching on original variables in Dantzig-Wolfe decompositions.

They argue that an original formulation can always be derived from a master and a pricing problem. Branching can thus be performed in the original formulation, which is then re-decomposed into an equivalent master and pricing problem. Branching rules on the original variables will be present in the master or in the pricing problem; either way, branching may change the pricing problem because of extra constraints or because of a changed reduced cost function.

Vanderbeck [20] presented a generic branching strategy for branch-and-price algorithms. The strategy consists of branching on disaggregated variables similar to Villeneuve et al. [21] but such that branching can always be imposed by fixing variable bounds in the pricing problem. This imposes only few changes to the pricing problem variable bounds, it allows for early pruning of parts of the sub tree, and it reduces symmetry in the branch-and-bound tree.

In this paper, we propose a general method for solving aggregated formulations based on applying Benders' decomposition [7] on a combination of the disaggregated and aggregated formulations. The proposed exact method ensures feasibility even when the aggregated formulation is a relaxation of the original problem, and it provides a finite branching strategy based on the original, disaggregated variables. The method also provides an LP bound, which is at least as good as those for the disaggregated and aggregated formulations.

The constraints of the disaggregated formulation and constraints linking the disaggregated and aggregated variables are added to the aggregated formulation. It is LP-relaxed and decomposed into a Benders' master and sub problem. The master problem is the aggregated formulation plus Benders' cuts. The Benders' dual sub problem is the disaggregated formulation and the linking constraints. Given a fractional solution optimal to the master problem, branching is imposed on the disaggregated variables in Benders' dual sub problem. In each branching child, the Benders' dual sub problem is resolved with the added branching constraint and the resulting Benders' cut is added to the master problem.

The Benders' master problem can be solved using branch-and-price. Then the proposed method does not change the solution polytope of the pricing problem, but instead modifies the reduced costs to reflect the branching strategy. This, and the fact that the method can handle relaxed aggregated formulations (i. e., aggregated formulations which do not exactly represent the original problem) compliment the work by Vanderbeck [20] and Villeneuve et al. [21]. Neither Villeneuve et al. or Vanderbeck consider relaxed aggregated formulations, and both their branching strategies potentially change the solution polytope of the pricing problem (but not necessarily the reduced costs).

Farka's Lemma has previously been used to derive cutting planes in an equivalent fashion to the Benders' cuts. Applegate et al. [2, 3] derive cuts for the travelling salesman problem by linking a path and and an edge formulation. The classical edge formulation is solved using a cutting plane approach and the fractional solution is mapped into a linear programming relaxed path formulation. An infeasible mapping of the solution implies that Farka's Lemma can be used to generate a cut using the dual ray. A similar approach is used for the vehicle routing problem by Ralphs and Galati [16] and by Ralphs et al. [17].

This paper is structured as follows. First in Section 2, we give an overview of the proposed solution approach. Then in Section 3, we prove the correctness of the method by applying it to a general framework. Section 4 analyzes drawbacks and benefits. Finally, Section 5 concludes the work.

2 Overview of exact solution method

The proposed solution method is two-fold. First a new formulation must be constructed from a disaggregated and an aggregated formulation, and Benders' decomposition must be applied on the new formulation. Then the actual solution procedure is begun, where the branching strategy is applied via Benders' dual sub problem.

The first procedure is outlined in Algorithm 1. Given is a disaggregated formulation, (P1), and an aggregated formulation, (P2). We generate formulation (P3) by including all constraints and variables of (P1) and (P2), constraints to link the variables of (P1) and (P2), and finally we apply the objective function of (P2). Formulation (P3) is LP-relaxed and Benders' decomposition is applied to project out the disaggregated variables.

Algorithm 1 Preparation procedure

1: (P1) is the disaggregated formulation
2: (P2) is the aggregated formulation
3: (P3) consists of (P2) with LP-relaxed variables, the constraints and variables of (P1), and of constraints linking the variables of (P1) and (P2).
4: LP-relax all variables in (P3) and denote the resulting formulation (LP3)
5: Apply Benders' decomposition on (LP3) to project out the disaggregated variables

The actual solution procedure is shown in Algorithm 2. Given are the master and dual sub problem for (P3) from Algorithm 1. As we have LP-relaxed formulation (P3), we use branch-and-bound (denoted b&b in the algorithm) to eventually reach an integer solution. In each branch-and-bound node, we solve the Benders' master and dual sub problem. If the

solution to the latter contains fractional variables, then branching nodes are generated. In each branching child, the branching rule is applied to the *disaggregated variables in the dual sub problem* and the Benders' cut resulting from the branching rule is added to the master problem in the branching child.

Algorithm 2 Solution procedure

```
1: The b&b tree root node consists of Benders' master and dual sub problem from Algorithm 1
2: while (An open b&b node exists) do
3:   Solve the Benders' master and dual sub problem
4:   Keep track of bounds and prune b&b tree accordingly
5:   if (a variable in the Benders dual sub problem solution is fractional) then
6:     Generate branching children
7:     for (each branching child) do
8:       Copy the Benders' master and dual sub problem from the parent node
9:       Impose branching constraint on disaggregated variable in Benders' dual sub problem
10:      Resolve the dual sub problem
11:      if an integer solution is reached then
12:        Go to line 19
13:      else
14:        Add any resulting Benders' cut to the master problem
15:        Add branching child to the b&b tree as an open node
16:      end if
17:    end for
18:  else
19:    A feasible solution is reached. Update bounds
20:    Close branch-and-bound node
21:  end if
22: end while
```

The interesting parts of the two procedures are the generation and decomposition of formulation (P3) (Step 3-5 in Algorithm 1) and the branching procedure (Step 5-17 in Algorithm 2). The remaining steps of the algorithms are textbook procedures.

3 Correctness of proposed solution procedure

In this section, we prove the correctness of the proposed solution procedure using a general framework. First we go through the preparation procedure in

Algorithm 1 and prove its correctness. Then follows the solution procedure in Algorithm 2.

The formulations in the following contain integer variables. The proposed work is trivially also applicable on mixed integer programs and on binary integer programs.

We first consider the disaggregated (original) formulation, which contains variables x^k defined on the index $k \in K$. The disaggregated formulation denoted (P1) is:

$$\min \sum_{k \in K} \tilde{c} x^k \tag{1}$$

$$\text{s.t.} \sum_{k \in K} A x^k \geq b \tag{2}$$

$$x^k \in \mathbb{Z}^n \qquad \forall k \in K \tag{3}$$

The objective function (1) minimizes the cost of the solution. Constraints (2) constrain the values of variables, and bounds (3) ensure that variables take on feasible integer values.

The problem stated in (1)-(3) can be reformulated using aggregated variables. Instead of variables x^k defined on index $k \in K$, we consider aggregated variables x. The aggregated formulation denoted (P2) is:

$$\min cx \tag{4}$$

$$\text{s.t. } Gx \geq f \tag{5}$$

$$x \in \mathbb{Z}^m \tag{6}$$

The objective function (4) minimizes the cost of the solution using the aggregated variables. Constraints (5) constrain the solution polytope similarly to constraints (2) from the previous formulation, and bounds (6) force variables to take on feasible values.

Definition 1. Consider the objective functions (1) and (4). Given the same solution polytope, the cost coefficients are defined such that the optimal solution values of (1) and (4) are equal.

Even though (P1) and (P2) represent the same problem, their polytopes may differ. Constraints (2) and (5) may constrain the problem differently, thus solutions to (P2) can be infeasible to (P1):

Proposition 1. Let $z(\text{P1})$ be an optimal solution value to formulation (P1) and $z(\text{P2})$ an optimal solution value to formulation (P2). Then:

$$z(\text{P1}) \ \geq \ z(\text{P2})$$

Proof. (P1) formulates the original problem, and (P2) contains all feasible solutions to the original problem. Taking Definition 1 into account gives the desired result. □

Definition 2. In the case where:

$$z(\text{P1}) \ > \ z(\text{P2})$$

for some instances, we say that (P2) is a *relaxation* of (P1)

Formulation (P2) may be attractive to consider, even when it is a relaxation of (P1). One reason is that (P1) may be undesirable due to symmetry in the solution polytope or due to large numbers of variables and constraints. Another reason is that (P2) may provide good lower bounds.

Even when (P2) gives better bounds and thus is beneficial to solve, we still need to consider if a solution is feasible to the original problem. We do that by linking the variables of the formulations. The linking constraints are essential to the solution method, thus they must be chosen carefully.

Definition 3. Let x_o^k and x_0 be variables in (P1) resp. (P2). The *linking constraints* are:

$$Hx_0 = \sum_{k \in K} Dx_0^k \tag{7}$$

such that

$$x^k \in \mathbb{Z} \Rightarrow x \in \mathbb{Z} \tag{8}$$

i. e. such that any feasible solution to (P1) can be transformed into a feasible solution to (P2).

How to formulate the linking constraints is highly problem dependent. Examples of typical linking constraints are:

- $x_0 = \sum_{k \in K} x_0^k$ for problems where the variables in the aggregated formulation only differ from those in the disaggregated formulation by an index. For example for the SDVRP: the disaggregated formulation contains a binary variable for each edge and vehicle indicating if the vehicle travels on the edge, while the aggregated formulation contains integer variables for each edge counting the number of vehicles traveling on the edge.
- $cx = \sum_{k \in K} \tilde{c} x^k$ can be used to link all problems according to Definition 1. Using the objective functions to link the formulations may not always be a good choice, though, since the derived Benders' cuts are unable to exploit any problem structure.

- $Hx_0 = \sum_{k \in K} Dx_0^k$ for a given property of the problem. For example consider the cutting stock problem where the disaggregated formulation contains arc-flow variables each representing the cut of a given item, while the aggregated formulation contains pattern-based variables representing cuts of a selection of items. The the left hand side of the equation above would sum over pattern variables containing a given item, and the right hand side would sum over arc-flow variables representing the cut of the given item.

Let (LP1) be the LP-relaxation of formulation (P1) and (LP2) the LP-relaxation of formulation (P2). Furthermore, let z(LP1) be an optimal solution value to (LP1) and z(LP2) an optimal solution value to (P2). Using the linking constraints, we want to consider the feasibility of solutions to (LP1) and to (LP2).

Proposition 2. Given a feasible solution to (LP1) the following holds true:

$$\left(x^k \in \mathbb{Z}^n,\ \forall k \in K\right) \Rightarrow x \in \mathbb{Z}^m$$

Proof. This trivially follows from Definition 3. □

Assumption 1. Given a feasible solution to (LP2), the following holds true:

$$x \in \mathbb{Z}^m \Rightarrow \sum_{k \in K} x^k \in \mathbb{Z}^n$$

When Assumption 1 holds true, then branching can be imposed on the variables of (LP2) to eventually reach a solution feasible to both (P2) and (P1). Branching can be performed using e. g. the methods proposed by Vanderbeck [20] or Villeneuve et al. [21]. The method proposed in this paper is still applicable and is beneficial in a branch-and-price context where the complexity of the pricing problem depends on variable bounds.

Assume that Assumption 1 does *not* hold true. In this case, integrality cannot be imposed directly on the x variables in (LP2). The proposed work could be especially beneficial in this case.

Proposition 3. If Assumption 1 does *not* hold true, then a feasible solution to (P2) is not necessarily feasible to (P1).

Proof. Given a solution to (P2) with feasible values for x, it is not clear how to reach feasible values for the variables x^k. Considering Proposition 1, we may even have that the solution is infeasible to (P1). □

Having considered the relationship between (P1) and (P2), the two formulations are now combined. We add the linking constraints (7) and the

constraints and bounds of (P1) to (P2). The resulting formulation is denoted (P3i):

$$\min\ cx \tag{9}$$
$$\text{s.t. } Gx \geq f \tag{10}$$
$$x \in \mathbb{Z}^m \tag{11}$$
$$Hx = \sum_{k \in K} Dx^k \tag{12}$$
$$\sum_{k \in K} Ax^k \geq b \tag{13}$$
$$x^k \in \mathbb{Z}^n \qquad \forall k \in K \tag{14}$$

Theorem 1. Consider the formulations (P1), (P2) and (P3i). Then:

$$z(\text{P1}) = z(\text{P3i})$$

Proof. (P2) contains all feasible solutions to the original problem, and (P1) formulates the original problem. Hence (P1) constrains the problem at least as much as (P2), see Proposition 1. Using Definition 1 we have that z(P3i) must be equal to z(P1). □

The integrality constraints on the aggregated variables in (P3i) are dropped. The resulting formulation is denoted (P3):

$$\min\ cx \tag{15}$$
$$\text{s.t. } Gx \geq f \tag{16}$$
$$Hx = \sum_{k \in K} Dx^k \tag{17}$$
$$\sum_{k \in K} Ax^k \geq b \tag{18}$$
$$x^k \in \mathbb{Z}^n \qquad \forall k \in K \tag{19}$$

Theorem 2. Consider the formulations (P1), (P2) and (P3). The following holds true:

$$z(\text{P1}) = z(\text{P3})$$

Proof. Given a solution with integer variables x^k then the linking constraints ensure that a solution can be transformed such that all the x variables are also integer – this is according to Definition 3. Using Theorem 1 gives us the desired result. □

Proposition 4. Let (LP3) be the LP-relaxation of (P3). Consider the LP-relaxed formulations (LP1), (LP2) and (LP3). Then:

$$z(\text{LP3}) = \max\{z(\text{LP1}), z(\text{LP2})\}$$

Proof. Given Definition 1 and the fact that all constraints of both (P1) and (P2) are included, then z(LP3) can never be smaller than z(LP1) or z(LP2). □

Benders' decomposition is applied on (LP3) to project out the x^k variables. From this we get the Benders' Master Problem (BMP):

$$\begin{aligned}
&\min\ cx \\
&\text{s.t. } Gx \geq f \\
&\quad u_r^i(b + Hx) \leq 0 && \forall i = 1, \ldots, R \\
&\quad x \text{ free}
\end{aligned}$$

with $u_r^i,\ i = 1, \ldots, R$ being the dual extreme rays of the Benders' dual sub problem and h and g being some coefficients given by the cut. Note that Benders' cuts based on dual extreme points are omitted from the (BMP) since the Benders dual sub problem is only solved to ensure feasibility.

Given a solution $\bar{x}$ to the (BMP), Benders' Dual Sub Problem (BDSP) is:

$$\begin{aligned}
&\min\ 0 \\
&\text{s.t. } H\bar{x} = \sum_{k \in K} Dx^k \\
&\quad \sum_{k \in K} Ax^k \geq b \\
&\quad x^k \text{ free} && \forall k \in K
\end{aligned}$$

This ends the correctness of Algorithm 1: we have proved that (P2) contains all feasible solutions of (P1) and given examples on how to link the two formulations. Finally, (P3) is proved to be correct for the original problem and Benders' decomposition is applied on the LP-relaxed (P3) to project out the disaggregated variables.

In the following we prove the correctness of the solution procedure in Algorithm 2. Focus is on proving that an optimal solution to (BMP), where all variables in (BDSP) are integer, is also optimal to the original formulation.

Corollary 1. The solution value of (BMP) is equal to that of (LP3):

$$z(\text{BMP}) = z(\text{LP3})$$

Proof. This follows from the Benders' decomposition principle [7]. □

The following is key to the proposed work, as it proves the actual correctness of the solution procedure.

Lemma 1. Any feasible solution to (BMP), where all variables x^k in the feasible corresponding (BDSP) are integer, is feasible to (P3).

Proof. Consider applying Benders decomposition on (P3). The Benders Master Problem would be similar to (BMP) and the Benders Sub Problem to (BDSP) but with integrality constraint on the x^k variables. □

Theorem 3. An optimal solution to (BMP) with integer variables x^k in the corresponding (BDSP), is optimal to (P1).

Proof. This follows trivially from Lemma 1 and Theorem 2. □

To obtain a valid integer solution to (P3), we embed (BMP) into a branch-and-bound scheme as shown in Algorithm 2. We branch on disjunctions in the x^k-variable space to reach feasible solutions for (P3). Given a variable x_i^k with fractional solution value v, the disjunctions are enforced in the (BDSP) as

$$x_i^k \leq \lfloor v \rfloor \vee x_i^k \geq \lceil v \rceil$$

and are therefore implicitly enforced in (BMP) using the Benders' cuts.

In this way we eventually reach an optimal solution to (BMP) with integer variables x^k in the corresponding (BDSP). From Theorem 3 we have that this solution is optimal to our original problem formulation (P1). The solution procedures in Algorithm 1 and 2 thus provide an exact approach for solving aggregated formulations.

4 Analysis of the proposed method

In the following we consider possible extensions of the proposed solution method.

4.1 Cutting planes

Cutting planes can be used to strengthen the formulations (P1), (P2) and (P3). The cuts may be generated iteratively in a cutting plane algorithm. Here we define the cuts to be valid inequalities, see Definition 8.1 in Wolsey [22]. Note that if put in a branch-and-bound context, then the cuts may only be valid in the current node and its sub tree. The following thus considers using cutting plane algorithms in the current branch-and-bound node and its sub tree.

Consider using a cutting plane algorithm for solving the disaggregated formulation (P1), the aggregated formulation (P2), the combined formulation (P3) or all three. Denote a cut for (P1)

$$\pi x^k \leq \pi_0 \tag{20}$$

a cut for (P2)

$$\lambda x \leq \lambda_0 \tag{21}$$

and a cut for (P3)

$$\gamma x^k + \mu x \leq \omega_0 \tag{22}$$

Definition 4. The cut (20) is a valid inequality to (P1), the cut (21) is a valid inequality to (P2), and the cut (22) is a valid inequality to (P3)

Corollary 2. It is feasible to both pre-generate or iteratively add cuts of type (20) to (P1), cuts of type (21) to (P2), and cuts of type (22) to (P3)

Proof. According to Definition 4, then (20) are valid inequalities to (P1), (21) are valid inequalities to (P2), and (22) are valid inequalities to (P3). The cuts can thus be included in their respective formulations from the beginning. They can always be added iteratively according to the cutting plane algorithm defined in Section 8.5 in Wolsey [22]. □

Assume that all cuts were pre-generated and added to (P1), (P2) resp. (P3). Formulation (P3) with cuts is denoted (P3c) and is defined as:

$$\begin{aligned}
&\min\ cx && && (23)\\
&\text{s.t. } Gx \geq f && && (24)\\
&\quad Hx = \sum_{k\in K} Dx^k && && (25)\\
&\quad \lambda x \leq \lambda_0 && && (26)\\
&\quad \sum_{k\in K} Ax^k \geq b && && (27)\\
&\quad \pi x^k \leq \pi_0 && && (28)\\
&\quad \gamma x^k + \mu x \leq \omega_0 && && (29)\\
&\quad x^k \in \mathbb{Z}^n && \forall k \in K && (30)
\end{aligned}$$

Corollary 3. The cuts (26), (28) and (29) are valid inequalities to (P3)

Proof. This follows trivially from Definition 4 and from Theorem 2 □

Applying Benders' decomposition on the formulation above gives the following Benders' Master Problem with cuts (BMPc):

$$\begin{aligned}
&\min\ cx && && \\
&\text{s.t. } Gx \geq f && && \\
&\quad \lambda x \leq \lambda_0 && && (31)\\
&\quad u_r^i(b + Hx) \leq 0 && \forall i = 1, \ldots, R && \\
&\quad x \text{ free} && &&
\end{aligned}$$

with $u_r^i,\ i = 1, \ldots, R$ being the dual extreme rays of the Benders' dual sub problem and h and g being some coefficients given by the cut.

Given a solution $\bar{x}$ to the master problem, the Benders' Dual Sub Problem with cuts (BDSPc) is:

$$
\begin{aligned}
&\min\ 0 \\
&\text{s.t. } H\bar{x} = \sum_{k \in K} Dx^k \\
&\quad \sum_{k \in K} Ax^k \geq b \\
&\quad \pi x^k \leq \pi_0 && (32) \\
&\quad \gamma x^k \leq \omega_0 - \mu\bar{x} && (33) \\
&\quad x^k \text{ free} && \forall k \in K
\end{aligned}
$$

Corollary 4. The cuts (26), (28) and (29) are valid inequalities to (P3). Furthermore, cuts (31) (equivalent to (26)) are valid to (BMPc) and cuts (32)-(33) (equivalent to (28)-(29)) are valid to (BDSPc)

Proof. The cuts (26), (28) and (29) are valid to (P3) according Corollary 3. Using the Benders' decomposition principle, cuts (31) are in (BMPc) because they are based on the aggregated variables. Similarly the cuts (32) and (33) are in (BDSPc) because they are (partly for (33)) based on the disaggregated variables, which are projected out. □

Theorem 4. The proposed solution approach allows usage of cutting planes to solve formulation (P1), (P2), (P3) or all three.

Proof. Corollary 4 shows that cuts added to (P1), (P2) and (P3) are valid inequalities to the proposed solution approach. Furthermore, according to Corollary 2, the cuts can be iteratively added in a cutting plane context. □

The proposed solution method thus allows including cuts valid for both the aggregated and disaggregated formulations. This potentially strengthens the formulations and thus improves the solution time.

4.2 Branching strategy

An important performance parameter for branch-and-bound algorithms is the size of the branch-and-bound tree. When branching on variables, the number of branches may be smaller for aggregated formulations than for disaggregated formulation, simply because the former contains fewer variables. We thus propose to use the proposed branching strategy with care. Branching strategies on the aggregated or disaggregated variables may exist, cutting off large parts of the solution polytope without complicating the problem structure. These strategies could be more attractive.

Recall that the proposed method does not require integral values for the aggregated variables. Also, recall that when the aggregated formulation is a relaxation of the disaggregated formulation, then having integral values for all aggregated variables does not guarantee a feasible or optimal solution to the original problem. We may still, however, be interested in integral values for aggregated variables as this could cut off large part of the solution space faster than only branching on disaggregated variables.

Consider the branching rule on aggregated variables:

$$\sum_{i \in \lambda} x_i = v \quad \text{vs.} \quad \sum_{i \in \delta} x_i = u \tag{34}$$

for given sets λ and δ and constants v and u. The branching cut (34) can be added to the (BMP) in each branching child. The branching cut can be viewed as a valid inequality to the problem in the resulting branch-and-bound node. According to Theorem 4 adding the branching cut is hence feasible.

Consider the branching rule on disaggregated variables

$$\sum_{i \in \lambda} x_i^k = v \quad \text{vs.} \quad \sum_{i \in \delta} x_i^k = u \tag{35}$$

for given sets λ and δ and constants v and u. The branching cut (35) is added to the (BDSP) in each branching child. Again, the branching cut can be viewed as a valid inequality to the problem in the resulting branch-and-bound node, and adding the branching cut is feasible according to Theorem 4. This branching strategy may also not guarantee an integer solution; however, it may still be useful to cut off large parts of the solution polytope.

4.3 Column generation

The (BMP) can be solved using column generation, where a pricing problem generates columns with negative reduced cost (or positive reduced cost for maximization problems). The reduced cost is calculated using the dual variable values of the current solution to (BMP). Define y_0 to be the non-negative duals of the constraints in (BMP) and y_i the non-positive duals of the Benders' cuts in (BMP). The reduced cost is:

$$c - Gy_0 - \sum_{i=1}^{R} (u_r^i H) y_i \leq 0$$

In the proposed method, branching rules are imposed by Benders' cuts in the (BMP). The dual variables, y_i, and the coefficients, $u_r^i H$, of these cuts thus influence the reduced costs, i.e., the Benders' cuts change the objective

of the pricing problem. The Benders' cuts do not, however, change the solution polytope of the pricing problem.

The proposed solution method must be used with care in column generation context, if changing the objective function of the pricing problem causes extra computational time. If the objective function of the pricing problem does not influence the computational time significantly, then the proposed method may be attractive.

When solving the aggregated formulation using column generation, then branching is often done by bounding variables values in the pricing problem. If this causes extra computational time, then the proposed solution method may be very beneficial to use as alternative, as it does not change the variable bounds in the pricing problem.

5 Conclusion

Aggregating formulations is often very desirable to reduce the number of variables and/or constraints. The resulting formulation may be easier to solve and provide good bounds. The formulation, however, may be a *relaxation* of the original problem. It may also cause complicated branching, because branching cannot always be performed on the aggregated variables.

We propose a generic solution method for aggregated formulations. The method consists of combining the aggregated and disaggregated formulations and adding variable linking constraints. The combined formulation is Benders' decomposed such that the resulting Benders' Master Problem (BMP) is equivalent to the LP-relaxed aggregated formulation plus possible Benders' cuts. The resulting Benders' Dual Sub Problem (BDSP) is equivalent to the LP-relaxed disaggregated formulation plus the linking constraints. If the optimal solution to the (BMP) results in fractional variables in the (BDSP), then we branch on the original, disaggregated variables in the (BDSP). Two branching children are generated: in each child a fractional variable in the (BDSP) is floored (first branching child) or ceiled (second branching child) and the (BDSP) is resolved. From this solution, we get a cut, which is added to the (BMP) in the branching child and the procedure repeats.

The paper proves the correctness of the proposed generic solution approach, and that the approach guarantees an LP bound at least as good as those for the disaggregated and aggregated formulations. The paper furthermore considers how to include cutting planes in both the aggregated and disaggregated formulations. Finally, it considers how to extend the method with advanced, problem specific solution approaches such as specialized branching strategies and column generation.

A relevant question to ask is, if it is more beneficial to repeatedly solve the (BDSP) than to solve the disaggregated formulation in a branch-and-bound scheme. The linking constraints restrict the solution polytope of

the (BDSP), potentially reducing both the amount of symmetry and the number of possible non-zero variables. It is important to carefully consider the linking constraint in order to restrict the solution space of the (BDSP) as much as possible.

References

[1] C. Alves and J. M. Valério de Carvalho. A stabilized branch-and-price-and-cut algorithm for the multiple length cutting stock problem. *Computer & Operations Research*, 35:1315–1328, 2008.

[2] D. Applegate, R. Bixby, V. Chvátal, and W. Cook. Tsp cuts which do not conform to the template paradigm. In *Computational Combinatorial Optimization*, pages 261–303. Springer, 2001.

[3] D. Applegate, R. Bixby, V. Chvátal, and W. Cook. Implementing the dantzig-fulkerson-johnson algorithm for large traveling salesman problems. *Mathematical Programming*, 97(91-153), 2003.

[4] G. Baier, E. Kohler, and M. Skutella. On the k-splittable flow problem. *Algorithmica*, 42:231–248, 2005.

[5] J. M. Belenguer, M. C. Martinez, and E. Mota. A lower bound for the split delivery vehicle routing problem. *Operations Research*, 48(5):801–810, 2000.

[6] G. Belov. *Problems, Models and Algorithms in One- and Two-Dimensional Cutting*. PhD thesis, Technischen Universitaet Dresden, Germany, 2003.

[7] J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.

[8] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8:101–111, 1960.

[9] M. Dror and P. Trudeau. Split delivery routing. *Naval Res. Logist.*, 37:383–402, 1990.

[10] M. Gamst and B. Petersen. Comparing branch-and-price algorithms for the multicommodity k-splittable maximum flow problem. *European Journal of Operational Research*, 217(2):278–286, 2012.

[11] P. C. Gilmore and R. E. Gomory. A linear programming approach to the cutting-stock problem - part ii. *Operations Research*, 11:863–888, 1963.

[12] M. Jin, K. Liu, and R. O. Bowden. A two-stage algorithm with valid inequalities for the split delivery vehicle routing problem. *International Journal of Production Economics*, 105(1):228–242, 2007.

[13] L. Kantorovich. Mathematical methods of planning and organising production. *Management Science*, 6(366-422), 1960.

[14] C.-G. Lee, M. A. Epelman, C. C. W. III, and Y. A. Bozer. A shortest path approach to the multiple-vehicle routing problem with split pick-ups. *Transportation Research Part B: Methodological*, 40(4):265–284, 2006.

[15] M. E. Lübbecke and J. Desrosiers. Selected topics in column generation. *Operations Research*, 53(6):1007–1023, 2005.

[16] T. Ralphs and M. Galati. Decomposition and dynamic cut generation in integer linear programming. *Mathematical Programming*, 106:261–285, 2006.

[17] T. Ralphs, L. Kopman, W. Pulleyblank, and L. Trotter. On the capacitated vehicle routing problem. *Mathematical Programming*, 94:343–359, 2003.

[18] J. Truffot and C. Duhamel. A branch and price algorithm for the k-splittable maximum flow problem. *Discrete Optimization*, 5(3):629–646, 2008.

[19] J. M. Valério de Carvalho. Exact solution of bin-packing problems using column generation and branch-and-bound. *Annals of Operations Research*, 86:629–659, 1999.

[20] F. Vanderbeck. Branching in branch-and-price: a generic scheme. *Math. Program. Ser. A*, 130:249–294, 2011.

[21] D. Villeneuve, J. Desrosiers, M. E. Lübbecke, and F. Soumis. On compact formulations for integer programs solved by column generation. *Annals of Operations Research*, 139(1):375–388, 2005.

[22] L. A. Wolsey. *Integer Programming*. Wiley-interscience publication, 1998.

Aggregating formulations is a powerful approach for transforming problems into taking more tractable forms. Aggregated formulations can, though, have drawbacks: some information may get lost in the aggregation and - put in a branch-and-bound context - branching may become very difficult and even cause an infeasible solution.
In this paper, we consider (mixed) integer program formulations and propose a method for ensuring an optimal solution to the original (disaggregated) problem using an aggregated formulation. The method is based on Benders' decomposition on a combination of the disaggregated mathematical formulation and the aggregated formulation. The method allows usage of relaxed aggregated formulations and enables branching on both aggregated and disaggregated variables. Also, the method guarantees an LP bound at least as good as those for the disaggregated and aggregated formulations.
The paper includes general considerations on types of problems for which the method is of particular interest. Furthermore, we prove the correctness of the procedure and consider how to include extensions such as cutting planes and advanced branching strategies.

ISBN 978-87-92706-95-9

DTU Management Engineering
Department of Management Engineering
Technical University of Denmark

Produktionstorvet
Building 424
DK-2800 Kongens Lyngby
Denmark
Tel. +45 45 25 48 00
Fax +45 45 93 34 35

www.man.dtu.dk